METHOD AND SYSTEM FOR FINGERPRINT TEMPLATE MATCHING

Anthony P. Russo

5   ## CROSS REFERENCE TO APPENDIX

Appendix A, which is a part of the present disclosure, is a listing of software code for embodiments of components of this invention, which are described more completely below.

10   A portion of the disclosure of this patent document contains material which is subject to copyright protection. The copyright owner has no objection to the facsimile reproduction by anyone of the patent document or the patent disclosure, as it appears in the Patent and

15   Trademark Office patent files or records, but otherwise reserves all copyright rights whatsoever.

FIELD OF THE INVENTION

This invention relates to fingerprint verification.

20   More particularly, this invention relates to methods for matching fingerprint templates and structures thereof.

BACKGROUND OF THE INVENTION

Biometric identification is used to verify the

25   identity of a person by digitally measuring selected features of some physical characteristic and comparing those measurements with those filed for the person in a reference database, or sometimes on a smart card carried by the person. Physical characteristics that are being

30   used include fingerprints, voiceprints, the geometry of the hand, the pattern of blood vessels on the wrist and on the retina of the eye, the topography of the iris of the

eye, facial patterns and the dynamics of writing a
signature and typing on a keyboard.

The fingerprint is one of the most widely used
physical characteristics in biometric identification.

5   Fingerprints are utilized as the most reliable means to
identify individuals because of two outstanding
characteristics; namely that they remain unchanged all
through life and differ from individual to individual.
Fingerprints consist of raised friction ridges of skin

10 separated by recessed valleys of skin. Fingerprint
"minutiae" are conventionally defined as ridge endings or
ridge bifurcations where a ridge splits into two ridges.

Since the matching of a fingerprint as an image can
require large file capacity to store the measured and

15 referenced features and complex computation to match
measured features with reference features, fingerprint
identification is carried out using the positional
relationship of features or minutiae that are extracted
from the fingerprint image. The minutiae representing a

20 fingerprint image are digitized and stored in a digital
memory, which may be a read-only memory, magnetic tape, or
the like. A fingerprint digitized in this manner may be
compared with reference fingerprints stored in the memory.
For the comparison to work, the reference fingerprint and

25 the measured fingerprint must be extracted, characterized,
and digitized so that the fingerprint templates contain
the same information and have the same format.

One application of fingerprint identification is to
use the fingerprint to unlock a smart card which often

30 contains encrypted information. Presently, a PIN
(personal identification number) is required to be entered
by the user before the encrypted information can be
extracted from the smart card and used. The use of a PIN

has many drawbacks. For example, the accounting or card issuing organization faces significant administrative cost in handling the secret codes and the card holders have to memorize the secret codes.

5   Some privacy laws, for example, the privacy laws in Europe, require sensitive information such as a reference fingerprint template to be stored on the smart card in a way that it cannot leave the card without first unlocking the card with a PIN. Therefore, for a fingerprint

10   matching scheme to be practical in Europe, the fingerprint template matching algorithm must be executed by a microprocessor in the smart card. Otherwise, the smart card must be unlocked by some other mechanism, such as a PIN, before the reference fingerprint template can be

15   read. The difficulty of executing conventional fingerprint template matching algorithm on a smart card is mainly due to the limited computational capabilities and memory of a conventional smart card. For example, a conventional smart card typically has less than 512 bytes

20   of RAM (with 256 bytes being typical) and between 1 Kilobyte and 16 Kilobytes of memory. An 8-bit RISC (reduced instruction set computer) microprocessor has a speed between 1 MegaHertz to 10 MegaHertz which is quite slow given the magnitude of the computations required to

25   complete a comparison between a measured fingerprint and a reference fingerprint. In effect, the hardware constraints prevent the use of fingerprint to unlock data from smart cards.

       In addition to hardware constraints, another

30   important design criterion is cost. U.S. Patent No. 4,582,985 (hereinafter, the '985 patent) entitled "Data Carrier", issued April 15, 1986, to Bo Löfberg and hereby incorporated by reference in its entirety, describes a

-3-

data carrier of a card type. According to the '985 patent, the data carrier includes a fingerprint verification device for carrying out the verification process. The verification device includes a sensor device for sensing a finger tip of the owner and obtaining the corresponding finger print line information. A specialized smart card must be used to accommodate the sensor device since a conventional smart card does not provide such accommodations. In addition, since the fingerprint template generation, storage, and comparison are done at the data carrier, the microprocessor of a conventional smart card is not adequate. Hence, a specialized smart card having a more capable processor must be used, thereby increasing cost.

Therefore, what is needed are systems and fingerprint template matching algorithms that can be executed by a microprocessor with low memory and low computational capacities, thereby keeping the cost of the smart card at an acceptable level.

SUMMARY OF THE INVENTION

In accordance with the present invention, a smart card verification system and a fingerprint template matching algorithm are provided. The fingerprint template matching algorithm is capable of being executed by a microprocessor with low memory and low computational capacities.

In one embodiment of this invention, a reference fingerprint template and a measured fingerprint template are generated from a reference fingerprint image and a fingerprint image to be verified, respectively. Each template comprises a plurality of data chunks, each data chunk representing a minutia and comprising a location, a

minutia angle and a neighborhood. In one embodiment, the
location is represented by two coordinates (x_j, y_j), the
coordinates having a center point (0,0) at the upper left
hand corner. The minutia angle θ_j is the angle between
5   the x-axis and a line tangential to the ridge line at the
minutia. In one embodiment, each coordinate and the
minutia angle are quantized to a selected number of bits.
In general, the amount of quantization is a function of
the available memory and the degree of accuracy desired.
10   The neighborhood is made up of a predetermined number
of neighbor minutiae which are selected for every minutia
extracted from a fingerprint image. Each neighbor minutia
is characterized by three positional parameters with
respect to the selected minutia. The positional
15 parameters include a distance and two angles.
     In one embodiment, an optional neighborhood boundary
is drawn around a randomly-selected minutia. In one
embodiment, if the number of neighbor minutiae within the
neighborhood boundary is less than the predetermined
20 number, all neighbor minutiae within the neighborhood
boundary are selected. In another embodiment, a
predetermined number of the closest neighboring minutiae
are selected. In yet another embodiment, a predetermined
number of neighbor minutiae giving the best spread around
25 the randomly selected minutia are selected. In one
embodiment, the minutiae having the farthest distance from
each other are selected. In another embodiment, minutiae
that are very close, e.g., less than approximately 10
pixels, to each other are not selected. In another
30 embodiment, an arbitrary one of the very close minutiae is
selected. In one embodiment, a quadrant is arbitrarily
drawn using the randomly-selected minutia as the center

point and a predetermined number of minutiae are selected from each quadrant.

An x-axis is drawn in the direction of the line tangential to the ridge at the randomly-selected minutia.
5 A y-axis, perpendicular to the x-axis, is drawn. The x-axis and the y-axis intersect at the randomly-selected minutia. Hence, the randomly-selected minutia now has a position of (0,0) and is designated as the "center minutia." A first line is drawn between the center
10 minutia and one of its neighbor minutiae. The first line has a distance d_i which is one of the positional parameters representing the neighbor minutia. An angle ϕ_i between the first line and the x-axis is the second positional parameter representing the neighbor minutia. A
15 second line is then drawn from the neighbor minutia to intersect the x-axis, in the direction of the line tangential to the ridge of neighbor minutia. The angle φ_i between the x-axis and the second line is the third positional parameter representing the neighbor minutia.
20   In one embodiment, each positional parameter is quantized to a selected number of bits. For example, each angle ϕ_i and φ_i is quantized to six bits and the distance d_i is quantized to five bits.

A data chunk from each fingerprint template is loaded
25 into a random access memory (RAM) to be compared. In one embodiment, the data chunks are sorted, but such sorting is optional. In one embodiment, the data chunks are sorted in accordance to their x-coordinates. In another embodiment, the data chunks are sorted in accordance to
30 their y-coordinates. In one embodiment, the data chunks in the reference template are sorted. In another embodiment, the data chunks in the measured template are

sorted. In yet another embodiment, both the measured template and the reference template are sorted.

Each of the characterization parameters, i.e., location, minutia angle, and neighborhood, of the measured data chunk to be compared is compared with its respective counterpart in the reference data chunk. The order of the comparison may be varied. For example, the location parameter in the x-coordinates of the measured minutiae are compared first to the corresponding x-coordinates of the reference minutiae. In another embodiment, location parameter in the y-coordinates of the measured minutiae are compared first to the corresponding y-coordinates of the reference minutiae. In one embodiment, the comparison is straight subtraction. If the difference for a parameter pair is equal to or is less than a respective predetermined tolerance, the parameters match. If all of the parameters match, the data chunks match. On the other hand, if one of the parameters fail to match, the data chunks do not match and the next set of data chunks are compared.

The data chunk comparison begins with the first reference data chunk and the first measured data chunk and terminates if the predetermined number of data chunk matches have been reached (i.e., the templates match) or if all of the data chunk combinations have been compared. In one embodiment, if the first reference data chunk and the first measured data chunk do not match, the next measured data chunk is loaded into the RAM to be compared with the first reference data chunk. The process continues until all the measured data chunks in the measured fingerprint template have been compared. The next reference data chunk and the first measured data chunk are then loaded into the RAM to be compared. The

-7-

process continues until all the reference data chunks have been compared. In another embodiment, if the first reference data chunk does not match the first measured data chunk, the second reference data chunk is loaded into the RAM to be compared with the first measured data chunk. The process continues until all of the reference chunks have been compared. The second measured data chunk and the first reference data chunk are then loaded into the RAM to be compared. The process continues until all of the measured data chunks have been compared.

In one embodiment, where the data chunks are sorted, the comparison between a reference data chunk and the remaining uncompared measured data chunks is terminated if the difference between the reference data chunk and the measured data chunk exceeds the tolerance. In one embodiment, the next comparison starts with the next reference data chunk and the first matching measured data chunk.

In the alternative embodiment, the comparison between a measured data chunk and the remaining uncompared reference data chunks is terminated if the difference between the measured data chunk and the reference data chunk exceeds the tolerance. In one embodiment, the next comparison starts with the next reference data chunk and the first matching measured data chunk.

In one embodiment, the neighbor minutiae in a neighborhood are sorted by distance d_i. The neighborhood comparison starts with the first neighbor in the reference neighborhood and the first neighbor in the measured neighborhood and terminates when the predetermined neighbor matches are reached (i.e., the neighborhoods match) or when all of the neighbor combinations have been compared. Each reference's positional parameter is

compared with its corresponding measured positional parameter. In one embodiment, the comparison is done with straight subtraction. If the difference meets a predetermined tolerance, the parameters match. If all of

5   the positional parameters in the neighbor minutia match, the neighbor minutiae match. If one of the positional parameters does not match, the neighbor minutiae do not match.

In one embodiment, the reference fingerprint template

10 is stored in a static memory in a smart card and the fingerprint template matching algorithm is executed in a microprocessor in the smart card. In one embodiment, the reference fingerprint template is generated and stored at a smart card reader.

15 The above-described fingerprint template matching algorithm may be executed by a microprocessor with low memory and low computational capacities because the matching algorithm does not require computational intensive operations and does not require comparisons that

20 exceeds 8 bits.

BRIEF DESCRIPTION OF THE DRAWINGS

FIG. 1 shows a smart card system having a smart card

25 reader for reading a smart card.

FIG. 2A shows a reference fingerprint template divided into data chunks.

FIG. 2B shows a measured fingerprint template to be compared divided into data chunks.

30 FIG. 3 shows how a minutia is represented.

FIG. 4A shows a circular neighborhood boundary.

FIG. 4B shows a square neighborhood boundary.

FIG. 4C shows a method of neighbor selection.

FIG. 4D shows an angle to be represented in complex domain.

FIG. 4E shows an embodiment of a method of neighbor selection.

FIG. 5 shows construction of positional parameters for a neighbor minutia.

FIG. 6 shows two neighborhoods with errors between them.

FIG. 7A shows a reference data chunk comprising location, minutia angle and neighborhood parameters.

FIG. 7B shows a measured data chunk comprising location, minutia angle and neighborhood parameters.

FIG. 7C shows a reference neighborhood having various positional parameters.

FIG. 7D shows a measured neighborhood having various positional parameters.

FIG. 8A shows a flowchart of a method of template comparison.

FIG. 8B shows a flowchart of another method of template comparison.

FIG. 9A shows a flowchart of a method of detailed data chunk comparison.

FIG. 9B shows a flowchart of a method of detailed data chunk comparison.

FIG. 9C shows a flowchart of a method of detailed data chunk comparison.

FIG. 10 shows a flowchart of a method of detailed neighborhood comparison.

FIG. 11A shows a flowchart of a method of detailed neighbor comparison.

FIG. 11B shows a flowchart of a method of detailed neighbor comparison, with the neighbors sorted.

FIG. 12 shows a flowchart of a method of fingerprint template comparison in a multiple user situation.

Use of the same reference numbers in different figures indicates similar or like elements.

5

DETAILED DESCRIPTION

The following description is meant to be illustrative only and not limiting. Other embodiments of this invention will be obvious in view of the following

10 description to those skilled in the semiconductor processing arts.

FIG. 1 shows one implementation of the present invention. FIG. 1 shows a smart card reader system having a smart card 100 and a smart card reader 110. Smart card

15 100 comprises a static memory 102 for storing a reference fingerprint template, a microprocessor 104 for executing a fingerprint template matching algorithm, and RAM (random access memory) 106. Smart card reader 110 has a RAM 112 for storing a measured fingerprint template, a

20 microprocessor 114, and a fingerprint sensor 116. In one embodiment, fingerprint sensor 116 is a capacitive fingerprint sensor of the kind described in U.S. Patent Application No. 08/735,100 filed December 15, 1995, and assigned to the same assignee as the present invention,

25 which is hereby incorporated by reference in its entirety. Smart card reader 110 can be a stand-alone unit at a point-of-sale or can be embedded in an ATM, for example.

When a user wishes to gain access to smart card 100, he/she inserts smart card 100 into smart card reader 110.

30 Smart card reader 110 then prompts the user to place his/her finger on fingerprint sensor 116. The prompt can be either visual or audible. Microprocessor 114 on smart card reader 110 then digitizes the electrical signal

measured at fingerprint sensor 116, creates a fingerprint image, extracts minutiae from the fingerprint image, characterizes each minutia, creates a neighborhood for each minutia, and generates a measured fingerprint

5   template from the characterization parameters and the neighborhood. The measured fingerprint template is then stored in RAM 112. Template construction is discussed in detail below.

Minutiae can be extracted from a fingerprint using

10   any known methods. For example, ridge skeletonization technique can be used. This technique uses an iterative process to thin a ridge line by discarding pixels that do not affect the continuity of the ridge line until a single pixel line is remaining, the end point of which is the

15   minutia. Other types of minutia extraction technique are described in, for example, U.S. Patent No. 4,646,352 entitled "Method and Device for Matching Fingerprints with Precise Minutia Pairs Selected From Coarse Pairs" issued February 24, 1987 to Asai et al. and U.S. Patent No.

20   4,135,147 entitled "Minutiae Pattern Matcher" issued January 16, 1979 to Riganati et al. The minutia extraction technique is not described in detail in this application because it is not part of the invention.

Smart card reader 110 then sends the measured

25   fingerprint template to smart card 100 through a communication channel 120 at a selected speed, typically 9.6 Kilobits/second, or 9600 baud. Microprocessor 104 in smart card 100 receives the measured fingerprint template from smart card reader 110, reads the reference

30   fingerprint template from static memory 102 into RAM 106 chunk by chunk as needed, and executes a fingerprint template matching algorithm to compare the two fingerprint templates.

A collection of data chunks representing a fingerprint, either reference or measured is referred to herein as a fingerprint template. A data chunk consists information of an extracted minutia it corresponds to, including location, minutia angle and neighborhood, which will be discussed below. FIGs. 2A, 2B, 3, 4A-4E, 5, 6 and 7A-7D show in detail how a fingerprint template is constructed. FIG. 2A shows a reference fingerprint template 200 having a plurality of data chunks 201, 202, 203, etc. Typically, reference fingerprint template 200 comprises 50 chunks, i.e., typically, 50 minutiae are extracted from a fingerprint image. In one embodiment, the minimum number of minutiae required is 10. The minimum number of minutiae is generally determined by the required matching accuracy. In one embodiment, the maximum number of minutiae allowed is 128. Number 128 is chosen as the maximum number of minutiae allowed because with a typical number of 50, 128 gives a sufficient range of the number of minutiae that may be extracted from a fingerprint image. Of course, the minimum and the maximum numbers of minutiae extracted may be modified to suit a particular application. Reference fingerprint template 200 is stored, for example, in static memory 102 on smart card 100.

FIG. 2B shows a measured fingerprint template 210 having a plurality of data chunks 211, 212, 213, etc. Similarly, measured fingerprint template 210 typically comprises 50 data chunks. Measured fingerprint template 210 is generated, for example, by smart card reader 110 and is stored in RAM 112 in smart card reader 110. The number of data chunks extracted from a measured fingerprint may be different from the number of data chunks extracted from a reference fingerprint due to

possible false minutiae. In addition, the number of data
chunks extracted from a measured fingerprint may be
different from the number of data chunks extracted from a
reference fingerprint because only a partial fingerprint
5   is read. However, a match can nevertheless be had if the
number of fingerprint data chunk matches satisfies a
predetermined number of fingerprint data chunk matches.
Therefore, the number of fingerprint data chunks in the
measured fingerprint template does not need to be the same
10   as the number of fingerprint data chunks in the reference
fingerprint template.

In one embodiment, all the data chunks in both
reference fingerprint template 200 and measured
fingerprint template 210 contain the same number of bits.
15   A data chunk is constructed for each and every minutia
extracted from a fingerprint image. Therefore, the number
of data chunks making up each fingerprint template is
equal to the number of minutiae extracted from a
fingerprint image. For example, if eighty minutiae are
20   extracted from a fingerprint image, the fingerprint
template will contain eighty data chunks.

FIG. 3 shows how each minutia is represented. An x-
axis and a y-axis are constructed so that the point (0,0)
is at the upper left hand corner of a fingerprint image.
25   The x-axis and the y-axis are drawn regardless of the
rotation or the translation of the fingerprint, as will be
discussed later. Each minutia in a fingerprint image is
characterized by location (x_j, y_j) and a minutia angle θ_j,
where j is the number of minutiae extracted from a
30   fingerprint image. Location (x_j, y_j) is the location of
minutia j with respect to x- and y-axis. Minutia angle θ_j
is the angle between the x-axis and a line tangential to a
ridge 301 where minutia j is extracted. Of importance,

-14-

the minutiae of both the reference fingerprint and the measured fingerprint must contain the same characterization parameters.

A neighborhood NH_j is constructed for each minutia j, as described below. FIGs. 4A-4E illustrate various ways of selecting neighbor minutiae in constructing a neighborhood NH_j for minutia j. A neighborhood is constructed for each and every minutia extracted from a fingerprint image. In one embodiment, a neighborhood boundary is drawn around minutia j. For example, a circle 300 of a fixed radius r, e.g., 90 pixels, is drawn using minutia j as the center point, as shown in FIG. 4A, to provide a neighborhood boundary. In an alternative embodiment, a box 310, representing the neighborhood boundary, is drawn using minutia j as the center point, as shown in FIG. 4B. Of course, any predetermined neighborhood boundary of any shape or size can be drawn.

A predetermined number of neighbor minutiae are selected from the enclosed area. In one embodiment, fifteen (15) minutiae (excluding minutia j) within the neighborhood boundary are selected as the neighbor minutiae for neighborhood NH_j. If the enclosed area contains less than the predetermined number of neighbor minutiae, all of the neighbor minutiae contained in the enclosed area are selected. It is noted that neighborhood NH_j contains multiple neighbor minutiae N_1-N_i where i is a predetermined number, so that there is enough information to make the neighborhood NH_j unique enough to meet the accuracy requirement. The number of the neighbors, in general, depends on the uniqueness of the neighborhood NH_j desired. For example, if only four neighbors N_1-N_4 are selected for a neighborhood, the probability of having equivalent or matching neighborhoods is much higher than

-15-

the corresponding probability associated with a
neighborhood of fifteen neighbors. Similarly, a
neighborhood having thirty neighbors would have a lower
probability of being equivalent to other neighborhoods.
5   The uniqueness of the neighborhood determines how often a
good fingerprint is rejected (i.e., false rejection rate)
and how often a bad fingerprint is accepted (i.e., false
acceptance rate). The number of the neighbor minutiae
also depends on the acceptable computational complexity or
10   speed that can be satisfactorily performed by a
microprocessor.

In one embodiment, the neighbor minutiae are selected
such as to maximize the spread around the center point,
i.e., minutia j. Referring to FIG. 4C, an x-axis is drawn
15   in the direction of the line tangential to a ridge line at
minutia j. A line is drawn from minutia j to a minutia
N_1, forming an angle σ_1 between the x-axis and the line.
Similarly, a line is drawn from minutia j to each of the
remaining minutiae within the enclosed area, each forming
20   an angle σ_k with the x-axis, where k is the number of the
minutiae within the enclosed area. A mean is calculated
from all the angles σ_k.

In one embodiment, the mean is calculated in a
complex domain since mean calculation in a degree domain
25   would be more complicated due to the cyclical nature of
the angles. A complex number C can be described as a sum
of its real and imaginary parts a and b, respectively,
i.e., C = a + ib where i is the square root of -1.

Referring to FIG. 4D, angles can be expressed in
30   terms of complex numbers by the following equality:

$$C_k = \cos(\sigma_k) - i\sin(\sigma_k)$$

where

$$\sigma_k = \tan^{-1}(b_k/a_k)$$

-16-

Thus, the real part of angle σ_k is $a_k = \cos(\sigma_k)$ and the imaginary part of angle σ_k is $b_k = \sin(\sigma_k)$. To get an average of all the angles σ_k, each angle σ_k is converted to its complex equivalent. The real part a_k and the

5   imaginary part b_k are then averaged separately. The averaged real part a_{avg} and the averaged imaginary part b_{avg} are then converted back to a real number representation σ_{avg}, for example, $a_{avg} = \dfrac{\sum_k (a_k)}{k}$, $b_{avg} = \dfrac{\sum_k (b_k)}{k}$, and $\sigma_{avg} = \tan^{-1}(b_{avg} / a_{avg})$.

10   In one embodiment, each angle σ_k is examined to determine how close it is to the average angle σ_{avg} by subtracting angle σ_k from average angle σ_{avg} to get a difference σ_{kdiff}. A predetermined number of minutiae that have the largest difference σ_{kdiff} are selected so that the

15   entire neighborhood has maximum standard angular deviation. It is noted that if there is a cluster of minutiae such as cluster 301 shown in FIG. 4C, the average would be closer to cluster 301. Thus the minutiae that are within cluster 301 would be closer to the mean. In

20   one embodiment, if the number of minutiae within the boundary is more than the predetermined number of neighbor minutiae, the minutiae within cluster 301 would not be selected. If one or more minutiae must be selected from cluster 301 to meet the predetermined number of neighbor

25   minutiae, the minutiae are selected randomly. In other words, the minutiae within cluster 301 are less likely to be selected than the minutiae outside of cluster 301.

        In one embodiment, if multiple minutiae, such as minutiae N_5 and N_6, have the same angles, the distances

30   between minutia N_5 and minutia j and between minutia N_6 and minutia j are used for a tie breaker, e.g., the shorter the distance, the more likely the minutia would be

selected.   In another embodiment, if a minutia, i.e.,
minutia N_3, is too close to minutia j, e.g., 10 pixels,
the minutia is not selected.   In yet another embodiment,
the neighbor minutiae that are furthest away (in distance)
5 from each other are selected.   In one embodiment, if two
or more neighbor minutiae are very close to each other,
e.g., 10 pixels, such as neighbor minutia N_5 and neighbor
minutia N_7, either both neighbor minutiae N_5 and N_7 are
eliminated or one of neighbor minutiae N_5 and N_7 is
10 arbitrarily selected.

   The neighborhood boundary is optional in selecting
neighbor minutiae for a neighborhood.   For example, the
neighbor minutiae are selected such as to maximize the
spread around the center point, i.e., minutia j, as
15 described above.   However, in this embodiment, all of the
minutiae extracted from the fingerprint are used to
calculate the mean.   All the other embodiments described
above are also applicable to neighbor minutiae selection
without a boundary.   In an additional embodiment, a
20 quadrant is arbitrarily drawn with minutia j as the center
point and a predetermined number of neighbor minutiae are
then selected from each quadrant, as shown in Fig. 4E.   In
general, any neighbor minutiae extraction method may be
used to extract the predetermined number i of neighbor
25 minutiae for constructing a neighborhood NH_j.

   After the neighbor minutiae are selected, a
neighborhood NH_j is constructed by deriving three
positional parameters for each neighbor minutia.   The
positional parameters are derived in reference to minutia
30 j.   These positional parameters include a distance d_i and
two angles ϕ_i and φ_i, where i is the number of neighbor
minutiae.

FIG. 5 shows how the positional parameters are derived. An x-axis is drawn in the direction of the line tangential to a ridge line at minutia j. A y-axis perpendicular to the x-axis is then drawn, intersecting the x-axis at minutia j. Hence, minutia j has a position of (0,0) and will be designated as the "center minutia N_0." A line 500 is drawn from center minutia N_0 to a neighbor minutia N_i. Line 500 has a distance d_i. An angle ϕ_i is created between the x-axis and line 500. Another line 501 is drawn from neighbor N_i in the direction of a line tangential to a ridge line for neighbor minutia N_i to intersect and extend through the x-axis. The angle between the extended line 501 and the x-axis is angle φ_i. Hence, neighbor minutia N_i may be represented by (d_i, ϕ_i, φ_i). Similarly, neighbor minutia N_1 is at a distance d_1 from center minutia N_0. An angle ϕ_1 is created between line 502 and the x-axis. Angle φ_1 is created between line 503 and the x-axis. Neighbor minutia N_2 is at a distance d_2 from center minutia N_0. An angle ϕ_2 is created between line 504 and the x-axis. An angle φ_2 is created between line 505 and the x-axis. The process is repeated for all i neighbor minutiae of center minutia N_0. Hence, for a neighborhood having 15 neighbors (i.e., i=15) each of the 15 neighbors N_1, N_2, N_3, ..., N_{15} is represented by (d_i, ϕ_i, φ_i). Importantly, distance d_i and angles ϕ_i and φ_i are independent of any rotation or translation of a fingerprint image because these are relative positions with respect to center minutia N_0. Hence, when a fingerprint image is rotated or translated, all the minutiae associated with that center minutia N_0 are translated and rotated in the same fashion and amount and their relative positions with respect to the center minutia N_0 remain the same.

In one embodiment, each of the parameters d_i, ϕ_i, and φ_i are quantized to a selected number of bits. Quantization allows the reduction of the number of bits to represent a value. More specifically, during

5   quantization, all the numbers that fall between two boundary values are discretized to a quantization value between the boundary numbers.

There are several approaches to derive the quantization value. In linear quantization, a

10 · quantization value can be derived by, e.g., a practical approach or a centroid approach. In the practical approach, the quantization value can be calculated by rounding the quotient of an equation, e.g., x/N, where x is the unquantized value and N is the quantization

15   interval, to the closest integer number. For example, if the unquantized value is 5 and the quantization interval is 2, the quantization value is obtained by rounding 5/2 = 2.5, or 3. Similarly, if the unquantized value is 5 and the quantization interval is 4, then quantization value is

20   obtained by rounding 5/4 = 1.25 or 1. In a centroid approach, the quantization value is computed as

$$C = \frac{\int_i^j x \, p(x) \, dx}{\int_i^j p(x) \, dx},$$

where C is the quantization value; i and j are the boundary numbers of the interval to be quantized; and p is

25   the probability of occurrence of each number x.

In non-linear quantization, a look-up table may be created to specify the quantization value for each unquantized value. For example, the lookup table may specify for unquantized values from 0 to 1, the

30   quantization value is 1; for unquantized valued from 1 to

3, the quantization value is 2; for unquantized values
from 3 to 6, the quantization value is 5; and so on. Non-
linear quantization has the advantage of setting the
quantization value in accordance to the errors associated
5 with each minutia. For example, for a given minutia, the
closer the neighbor minutia is to the given minutia, the
smaller the error. Similarly, the further the neighbor
minutia, the larger the error. Therefore, for neighbor
minutiae closer to the given minutia, the parameters
10 associated with the neighbor minutia needs to be quantized
more finely than the neighbor minutiae that are further
away.

Although any amount of quantization may be used, the
amount of quantization depends on the preciseness of the
15 match desired and the memory space available. As is well
known in the art, the more the quantization, the less the
preciseness of the representation and the less the storage
space is required. For example, for a set of unquantized
values 1 through 10, 4 bits are required to represent each
20 number. If the quantization interval is 2, then the
largest quantization value needs to be represented is 10/2
or 5. Hence, all 10 numbers can be represented by 3 bits,
requiring less storage space. Similarly, if the
quantization interval is 5, the largest quantization value
25 needs to be represented is 10/5 or 2. Hence, unquantized
values 1 through 10 can now be represented by 2 bits,
requiring even less storage space. However, the largest
error in the first example would be 1 and the largest
error in the second example would be 2.5. Hence, the more
30 the quantization, the less the storage requirement and the
larger the error.

In one embodiment, distance d_i is quantized to five bits. In another embodiment, angles ϕ_i and φ_i are quantized to six bits each.

FIG. 6 illustrates two neighborhoods with slight
5   errors, both rotational and translational. Reference neighborhood 510 is constructed with a center minutia N_0 and two neighbor minutiae N_1 and N_2. Measured neighborhood 512 is constructed with a center minutia N_0' and two neighbor minutiae N_1' and N_2'. As can be seen,
10   reference neighborhood 510 and measured neighborhood 512 show some resemblance, i.e., neighbor minutiae N_1 and N_2 and N_1' and N_2', respectively, have similar positional relationships with respect to their center minutiae N_0 and N_0', respectively. However, there are errors between the
15   two neighborhoods. For example, there are slight differences in the distances as well as the angles. The slight differences may be due to the fact that a finger has a fluid surface, thus the surface of the finger stretches and varies under different conditions. Hence,
20   these two slightly varied neighborhoods may be the fingerprint from the same finger, but created at different times. The fingerprint matching algorithm, therefore, must take these slight variations into account because if a perfect match is required, the numerous erroneous
25   rejections will render fingerprint identification impractical.

Because matching tolerances must be accepted, the errors caused by a certain amount of quantization becomes unimportant in the resulting accuracy of the comparison.
30   For example, if the tolerance of distance d_i is set to 1 and the maximum error caused by quantization of distance d_i is also 1, if both the reference distance and the measured distance falls into the same quanta, the

distances match even though the actual distances may be
different. The acceptable tolerance, therefore,
determines how much quantization can be done for each
parameter.

5   Another advantage for quantization is to increase the
data transmission speed because the size of the data
packets becomes smaller after quantization. In one
embodiment, the quantized values are compressed using
well-known compression method such as, but not limited to,
10   run-length encoding or L-Z-W (Lemtel-Ziv-Welch)
compression. In the run-length encoding, for example, if
there are five number zero in a row, instead of storing
the numbers as "0 0 0 0 0," the numbers are stored as "5
0." In the L-Z-W compression, a dictionary of frequently
15   repeated groups of (8-bit) characters is built on a per-
file basis in order to represents frequent character-
strings as shorter bit patterns. Hence, compression
allows the data to be represented by fewer bits, thus
allowing the data transmission to be faster and data
20   storage to be smaller. In one embodiment, the compressed
values are transmitted from smart card reader 110 into
smart card 100 and are decompressed at smart card 100.

In one embodiment, the location information, i.e.,
(x_j, y_j) and the minutia angle θ_j are quantized to a
25   selected number of bits. For example, each of the
position measurements x_j and y_j is quantized to four bits
and the minutia angle θ_j is quantized to six bits.

The size of a data chunk is determined by the size of
the parameters representing the location of the minutia,
30   the minutia angle, and its neighborhood. In one
embodiment, each data chunk in the fingerprint template is
34 bytes in size. In this embodiment, the fingerprint
template is made of 1-byte location information (i.e., two

4-bit measurements x_j and y_j), 6-bit minutia angle θ_j information, and fifteen 17-bit (5 bits for d_i, 6 bits each for angles ϕ_i and φ_i) neighborhood information.

As will be seen later, the fingerprint template matching algorithm compares the fingerprint templates on a chunk-by-chunk basis. Hence, the fingerprint template matching algorithm requires RAM 106 to be of a size large enough to store at least two data chunks, e.g., 68 bytes of data. In a conventional smart card where the typical RAM is 256 bytes in size, no additional memory is required because the required memory in accordance with this invention is within, in fact, well under, the memory confinement of a conventional smart card.

FIG. 7A shows reference data chunk 201 of reference fingerprint template 200 in detail. Reference data chunk 201 includes three parameters: location 702, minutia angle 704, and neighborhood 706. Location 702 includes x- and y- coordinates. Neighborhood 706, in one embodiment, includes 15 neighbor minutiae N_i through N_{15}. Each neighbor N_1 through N_{15} further includes positional parameters such as distance d_i between the center minutia N_0 and the neighbor minutia N_i and two related angles ϕ_i, φ_i, where i is 1 through 15, as shown in FIG. 7C.

Similarly, FIG. 7B shows data chunk 211 of measured fingerprint template 210 in detail. Measured data chunk 211 includes three parameters: location 752, minutia angle 754, and neighborhood 756. Location 752 includes x- and y-coordinates x_j' and y_j'. Neighborhood 756, in one embodiment, includes 15 neighbors N_1' through N_{15}'. Each neighbor N_1' through N_{15}' further includes positional parameters such as distance d_i' between the center minutia N_0' and a neighbor N_i' and two related angles ϕ_i', φ_i', where i is 1 through 15, as shown in FIG. 7D. Although

-24-

FIGs. 7A and 7B show the same number of neighbors in reference fingerprint data chunk 201 and measured fingerprint data chunk 211, it is not necessary because a neighborhood match may be found if the number of neighbor

5   minutia matches satisfies the predetermined number of neighbor minutia matches. For example, for a certain neighborhood boundary drawn, the enclosed area at the reference fingerprint contains enough minutiae to select the predetermined number of neighbor minutiae, e.g., 15

10 minutiae. However, the enclosed area at the measured fingerprint only contains less than the predetermined number of neighbor minutiae, e.g., 12 minutiae. If the predetermined neighbor minutia match is 7 and there are 7 matches between the measured neighbor minutiae and the

15 reference neighbor minutiae, there is a match between the reference neighborhood and the measured neighborhood even though neighborhood 706 contains different number of neighbors from neighborhood 756.

　　　FIG. 8A illustrates the fingerprint template matching

20 process in a smart card application. It is noted that although a fingerprint template matching algorithm is described with respect to a smart card and a smart card reader, the fingerprint template matching algorithm described hereinafter can also be executed by, for

25 example, a personal computer, a microprocessor system, or a palm computer.

　　　The template matching process starts with step 600. In one embodiment, the reference data chunks and the measured data chunks are sorted (step 601), but the

30 sorting is not required. In one embodiment, the data chunks are sorted according to its location information which is represented by the x- and y-coordinates. For example, the data chunks are sorted according to

increasing x-coordinate x_j followed by increasing y-coordinate y_j. Of course, any sorting technique can be used to sort the data chunks. In general, minutia angle θ_j is not used for sorting due to the cyclical nature of

5 angle measurements. In one embodiment, either the reference data chunks or the measured data chunks are sorted. In another embodiment, both the reference data chunks and the measured data chunks are sorted. The data chunks are sorted for efficiency purpose, as will be

10 explained below.

Because the fingerprint templates are compared on a chunk-by-chunk basis and the fingerprint templates are serialized, no random access to the fingerprint templates is needed. In other words, once a data chunk has been

15 used, that data chunk is discarded from RAM 106 and a new data chunk is read in. This allows the storage of only one data chunk from each template in RAM 106 during the comparison process.

The first data chunk of measured fingerprint template

20 210, e.g., measured chunk 211, is loaded from, for example, RAM 112 in smart card reader 110 into RAM 106 (step 602) through communication channel 120 (also refer to FIG. 1). Also in step 602, the first data chunk of reference fingerprint template 200, e.g., reference chunk

25 201, is loaded from static memory 102 into RAM 106 of smart card 100. Microprocessor 104 compares the two data chunks, i.e. measured chunk 211 and reference chunk 201 in RAM 106 (step 604). How the data chunks are compared in step 604 is discussed in detail later with reference to

30 FIGs. 9A, 9B and 9C.

If the two fingerprint template data chunks, e.g., reference chunk 201 and measured chunk 211, match (step 606), a counter is incremented in step 608. If the number

of fingerprint data chunk matches is equal to or greater than a predetermined number (step 610), measured fingerprint template 210 is considered to match reference fingerprint template 200 (step 612) and the fingerprint

5   template matching process terminates in step 614. In one embodiment, more than four data chunk matches is considered a match.

If the two fingerprint template data chunks, e.g., reference chunk 201 and measured chunk 211, do not match

10   (step 606) or if the number of data chunk matches is less than the predetermined number (step 610), it is determined whether the measured chunk used for the comparison, e.g., measured chunk 211, is the last measured chunk in measured fingerprint template 200 (step 616).

15   If the measured chunk, e.g. measured chunk 211, is not the last measured chunk in measured fingerprint template 210, the next measured chunk, e.g., measured chunk 212, in measured fingerprint template 210 is loaded into RAM 106 from RAM 112 via communication channel 120

20   (step 618). The newly loaded measured chunk, e.g., measured chunk 212, is compared with the reference chunk in RAM 106, e.g., reference chunk 201 (step 604). The process continues until all measured chunks in measured fingerprint template 210 have been compared or until a

25   predetermined number of data chunk matches have been found.

If the measured chunk, e.g., measured chunk 201, is the last reference chunk in measured fingerprint template 210, then it is determined whether the reference chunk in

30   RAM 106 used for comparison, e.g., reference chunk 201, is the last data chunk in reference fingerprint template 210 (step 620).

-27-

If the reference chunk, e.g., reference chunk 201, is
not the last reference chunk in reference fingerprint
template 200, the next reference chunk in reference
fingerprint template 200, e.g., reference chunk 202, is

5   loaded, for example, from static memory 102 into RAM 106
(step 622). Also in step 622, the first measured chunk in
fingerprint template 210, e.g., measured chunk 211, is
loaded into RAM 106. The newly loaded reference chunk,
e.g., reference chunk 202, and first measured chunk, e.g.,

10   measured chunk 211, are then compared in step 604. The
process is continued until all reference chunks in
reference fingerprint template 200 are compared or until
the number of data chunk matches exceeds the predetermined
number.

15   If the reference chunk, e.g., reference chunk 201, is
the last reference chunk in reference fingerprint template
200, reference fingerprint template 200 and measured
fingerprint template 210 do not match (step 624) and the
fingerprint template matching process terminates in step

20   614.
Alternatively, the fingerprint template matching
process can be carried out in accordance with FIG. 8B. In
FIG. 8B, the optional sorting is eliminated; the reference
chunks are rotated first; the measured chunks are then

25   rotated. In particular, the fingerprint template matching
process starts at step 650. The first data chunk of
measured fingerprint template 210, e.g., measured chunk
211, is loaded from smart card reader 110 into RAM 106
(step 652) through communication channel 120. Also in

30   step 652, the first data chunk of reference fingerprint
template 200, e.g., reference chunk 201, is loaded from
static memory 102 into RAM 106 of smart card 100.
Microprocessor 104 compares the two fingerprint template

-28-

data chunks, e.g., measured chunk 211 and reference chunk 201 in RAM 106 (step 654). How the data chunks are compared in step 654 is discussed in detail below with reference to FIG. 9C.

5   If the two data chunks, e.g., chunks 201 and 211, match (step 656), a counter is incremented in step 658. If the number of data chunk matches is equal to or is greater than a predetermined number (step 660), measured fingerprint template 210 matches reference fingerprint

10   template 200 (step 662) and the process ends in step 664.

On the other hand, if the two data chunks, e.g., data chunks 201 and 211 do not match (step 656) or if the number of matches of the data chunks is less than the predetermined number (step 660), it is determined whether

15   the measured chunk used for comparison, e.g., reference chunk 201, is the last reference chunk in reference fingerprint template 200 (step 666).

If the reference chunk, e.g., reference chunk 201, is not the last data chunk in reference fingerprint template

20   200, the next reference chunk, e.g., reference chunk 202, in reference fingerprint template 200 is loaded from static memory 102 into RAM 106 (step 668). The newly loaded reference chunk, e.g., reference chunk 202, is compared with the measured chunk in RAM 106, e.g.,

25   measured chunk 211, in step 654. The process continues.

If the reference chunk, e.g., reference chunk 201, is the last reference chunk in reference fingerprint template 200, it is determined whether the measured chunk in RAM 106 used for comparison, e.g., measured chunk 211, is the

30   last measured chunk in measured fingerprint template 210 (step 670).

If the measured chunk, e.g., measured chunk 211, is not the last data chunk in measured fingerprint template

-29-

210, the next measured chunk in measured fingerprint
template 210, e.g., measured chunk 212, is loaded from RAM
112 into RAM 106 of smart card 100 via communication
channel 120 (step 672). The first reference chunk in

5   fingerprint template 200, e.g., reference chunk 201, is
also loaded into RAM 106 (step 672). The newly loaded
measured chunk, e.g., measured chunk 212, and the first
reference chunk, e.g., reference chunk 201, are then
compared (step 654).

10   If the measured chunk, e.g., measured chunk 211, is
the last measured chunk in measured fingerprint template
210, measured fingerprint template 210 and reference
fingerprint template 200 do not match (step 674) and the
fingerprint template matching process terminates at step

15   664.
          FIG. 9A illustrates a flowchart of step 604 of how
the fingerprint template data chunks are compared in
detail. A reference chunk includes a positional parameter
(x_j, y_j) where x_j is the x-coordinate and y_j is the y-

20   coordinate of a reference minutia. A measured chunk
includes a positional parameter of (x_j', y_j') where x_j' is
the x-coordinate and y_j' is the y-coordinate of a measured
minutia. In this embodiment, the measured data chunks are
sorted in accordance with x-coordinates x_j followed by y-

25   coordinates y_j. In one embodiment, the x-coordinates are
compared by subtracting measured x-coordinate x_j' from
reference x-coordinate x_j, i.e. $x_j - x_j' = x_{jdiff}$ (step 850).
If the absolute value of difference x_{jdiff} is greater than
the predetermined tolerance $x_{jtolerance}$ (step 852), then it

30   is determined whether the raw difference x_{jdiff} is less
than the predetermined tolerance $x_{jtolerance}$ (step 868). If
difference x_{jdiff} is less than $x_{jtolerance}$, the rest of the
measured data chunks will not be compared against the

reference data chunk and it is determined whether the reference chunk used is the last reference chunk (step 620 in FIG. 8A) and the process continues. If the difference x_{jdiff} is greater than $x_{jtolerance}$, the chunks do not match

5 (step 872) and the process continues at step 606 (FIG. 8A).

If the absolute value of the difference x_{jdiff} is less than or equal to a predetermined tolerance $x_{jtolerance}$ (step 852), i.e., the x-coordinates match, the y-coordinates are

10 compared. Measured y-coordinate y_j' is subtracted from reference y-coordinate y_j, i.e. $y_j - y_j' = y_{jdiff}$ (step 854). If the absolute value of the difference y_{jdiff} is greater than the predetermined tolerance $y_{jtolerance}$ (step 856), it is determined whether the raw difference y_{jdiff} is less

15 than $y_{jtolerance}$. If difference y_{jdiff} is less than $y_{jtolerance}$, the remaining measured data chunks will not be compared against the reference data chunk and it is determined whether the reference chunk used for comparison is the last reference chunk (step 620 in FIG. 8A). If difference

20 y_{jdiff} is greater than $y_{jtolerance}$, the chunks do not match (step 872) and the process continues at step 606 (FIG. 8A). If the absolute value of the difference y_{jdiff} is less than or equal to a predetermined tolerance $y_{jtolerance}$ (step 856), i.e., the y-coordinates match, the minutia

25 angles θ_j and θ_j' are compared.

It is noted that a straight subtraction in degree domain does not work well with angles due to their cyclical nature. Measured minutia angle θ_j' is subtracted from reference minutia angle θ_j, i.e. $\theta_j - \theta_j' = \theta_{jdiff}$ (step

30 858). If the absolute value of the difference θ_{jdiff} is less than or equal to a predetermined tolerance $\theta_{jtolerance}$ (step 860), i.e., the minutia angles match. If the absolute value of the difference θ_{jdiff} is greater than the

-31-

predetermined tolerance $\theta_{jtolerance}$, it is determined whether
the absolute value of the difference θ_{jdiff} is greater than
or equal to the difference between the maximum allowable
quantized minutia angle θ_{jmax} and the predetermined

5   tolerance $\theta_{jtolerance}$ (step 861). If value $|\theta_{jdiff}|$ is greater
than or equal to the difference between values θ_{jmax} and
$\theta_{jtolerance}$, the minutia angles match.

An example for angle comparison is described below.
In the embodiment where the angles are quantized to 6

10   bits, thus having values 0 to 63, 0 is the minimum value
for θ_j (θ_{jmin}) and 63 is the maximum value for θ_j (θ_{jmax}).
The quantized values are then subtracted, e.g., $|\theta_j -$
$\theta_j'| = \theta_{jdiff}$, if minutia angle θ_j has a quantized value of 61
while minutia angle θ_j' has a quantized value of 63, and

15   the match tolerance is 2. Then $|61-63|=2$, which is within
the tolerance range. Thus, there is a match between
minutia angles θ_j and θ_j'. In a situation where angle θ_j
has a quantized value of 0 while minutia angle θ_j' has a
quantized value of 63, and the match tolerance is 2. Then

20   $|0-63|=63$, which is out of the tolerance range. However,
due to the cyclical nature, the difference between 63 and
0 is actually only 1, which is within the tolerance.
Therefore, in addition to checking whether $|\theta_j -$
$\theta_j'| \leq \theta_{jtolerance}$, whether $\theta_{jdiff} \geq \theta_{jmax} - \theta_{jtolerance}$ is also checked.

25   In the above example, $\theta_{jdiff} \geq \theta_{jmax} - \theta_{jtolerance}$ ($63 \geq 63-2$) is
true. Therefore, minutia angles θ_j and θ_j' match.

If the minutia angles θ_j and θ_j' match, neighborhood
NH_j is compared with neighborhood NH_j' in step 862, which
is described in detail in reference to FIG. 10.

30   If the neighborhood NH_j from the reference chunk
matches neighborhood NH_j' from the measured chunk (step
864), the data chunks match (step 866). Conversely, if
any of the parameters, i.e. x-coordinates, y-coordinates,

minutia angles and neighborhoods, fail to match, the data chunks do not match (step 872). The typical value for $x_{jtolerance}$ and $y_{jtolerance}$ is 4 and the typical value for $\theta_{jtolerance}$ is 7. These typical values are derived from

5   experimentation.

Since the measured data chunks are sorted in accordance with the x-coordinates x_j, the comparisons for the remaining measured data chunks can be eliminated because the remaining measured data chunks will also have

10   x-coordinates x_j' exceeding the tolerance. Hence, sorting cuts down the number of comparisons. The number of comparisons can be further reduced by sorting the data chunks by y-coordinates y_j.

A flag can be set so that only a subset of the

15   measured data chunks is used for further comparisons. FIG. 9B illustrates a flowchart of step 604 of how the fingerprint template data chunks are compared in detail. The x-coordinates are compared by subtracting measured x-coordinate x_j' from reference x-coordinate x_j, i.e. $x_j -$

20   $x_j' = x_{jdiff}$ (step 880). If the absolute value of the difference x_{jdiff} is less than or equal to a predetermined tolerance $x_{jtolerance}$ (step 881), it is determined whether this is the first match between reference x-coordinate x_j and measured x-coordinate x_j' (step 885). If the match is

25   the first match, a flag is set for the measured chunk containing the first matching x-coordinate x_j' (step 886).

If the absolute value of the difference x_{jdiff} is greater than the predetermined tolerance $x_{jtolerance}$ (step 881), then it is determined whether the difference x_{jdiff}

30   is less than the predetermined tolerance $x_{jtolerance}$ (step 882). If difference x_{jdiff} is less than $x_{jtolerance}$, the rest of the measured data chunks will not be compared against the reference data chunk and it is determined whether the

-33-

reference chunk used is the last reference chunk (step
883). If the reference chunk is not the last chunk in the
reference fingerprint template, the next reference chunk
and the start chunk is loaded into the RAM for comparison
5 (step 884) and the process continues with step 880. If
the reference chunk is the last chunk in the reference
fingerprint template, the fingerprint templates do not
match (step 624 in FIG. 8A).

     If the match is not the first match (step 885) or
10 after the start chunk has been flagged (step 886), the y-
coordinates are compared. Measured y-coordinate y_j' is
subtracted from reference y-coordinate y_j, i.e. $y_j - y_j' = y_{jdiff}$ (step 887). If the absolute value of the
difference y_{jdiff} is greater than the predetermined
15 tolerance $y_{jtolerance}$ (step 888), it is determined whether
the difference y_{jdiff} is less than $y_{jtolerance}$ (step 889). If
difference y_{jdiff} is less than $y_{jtolerance}$, the remaining
measured data chunks will not be compared against the
reference data chunk and it is determined whether the
20 reference chunk used for comparison is the last reference
chunk (step 883). If difference y_{jdiff} is greater than
$y_{jtolerance}$, the chunks do not match (step 896) and the
process continues at step 606 (FIG. 8A).

     If the absolute value of the difference y_{jdiff} is less
25 than or equal to a predetermined tolerance $y_{jtolerance}$ (step
888), the remaining steps 890 through 896 are similar to
steps 858-866 described in reference to FIG. 9A.

     As can be seen from FIGs. 9A and 9B, x-coordinates
and y-coordinates comparison filters the number of
30 comparisons. Although the fingerprint template matching
algorithm is capable of comparing a fingerprint with a
rotated and/or translated same fingerprint, the algorithm
can execute faster if the finger is constrained in a

-34-

certain orientation. By allowing translation and
rotation, none, or very few values can be eliminated
because the tolerance must be sufficiently large so as not
to eliminate everything during the comparison and thus

5   generating a false rejection.

FIG. 9C illustrates a flowchart of step 654 of how
the fingerprint template data chunks are compared in
detail, where every chunk is compared. Parameters x_j and
x_j' are compared (step 806), e.g., by straight

10   subtractions. If value x_{jdiff}, i.e., $|x_j-x_j'|=x_{jdiff}$, is
greater than a tunable, predetermined tolerance $x_{jtolerance}$,
parameter x_j does not match parameter x_j'. Therefore, the
data chunks do not match (step 824) and the next data
chunk is read (refer to FIG. 8B). When value x_{jdiff} is

15   equal to or is less than the tolerance $x_{jtolerance}$ (step
812), parameter x_j matches parameter x_j' (step 808) and
parameter y_j is compared with parameter y_j' in step 810
with a straight subtraction.

If value y_{jdiff}, i.e. $|y_j'-y_j|=y_{jdiff}$, is greater than a

20   tunable, predetermined tolerance $y_{jtolerance}$, parameter y_j
does not match parameter y_j' (step 812) and the data
chunks do not match (step 824). The next data chunk is
read (see FIG. 8B). When value y_{jdiff} is equal to or is
less than the tolerance $y_{jtolerance}$, parameter y_j matches

25   parameter y_j' (step 812) and parameter θ_j is compared with
parameter θ_j' in step 814.

If minutia angle difference θ_{jdiff} is greater than a
tunable, predetermined minutia angle tolerance $\theta_{jtolerance}$,
parameter θ_j does not match parameter θ_j' (step 816) and

30   the data chunks do not match (step 824) and the next data
chunk is read (see FIGs. 8A and 8B). When minutia angle
difference θ_{jdiff} is equal to or is less than the minutia
angle tolerance $\theta_{jtolerance}$ or when difference θ_{jdiff} is

-35-

greater than or equal to the maximum allowable quantized minutia angle minus the predetermined tolerance $\theta_{jtolerance}$, parameter θ_j matches parameter θ_j' (step 816) and neighborhood NH_j is compared with neighborhood NH_j' in step 818, which is described in detail in reference to FIG. 10. If the neighborhood NH_j from the reference chunk matches neighborhood NH_j' from the measured chunk (step 820), the data chunks match (step 822). Conversely, if the neighborhood NH_j and neighborhood NH_j' do not match, the data chunks do not match. The sequence of comparison may vary. For example, parameter y_j maybe compared before comparing parameter x_j; angle θ_j maybe compared before comparing parameter y_j; and so on.

FIG. 10 shows a flowchart of the neighborhood matching algorithm of steps 862, 893 and 818. In step 900, the first neighbor in an reference chunk (hereinafter reference neighbor), e.g., N_1 (FIG. 7A), and the first neighbor in a measured chunk (hereinafter measured neighbor), e.g., N_1' (FIG. 7B), are compared. Step 900 is described in detail below.

If the neighbors match (step 902), a neighbor counter is incremented in step 904. If the counter value is equal to or is greater than a predetermined number of neighbor matches (step 906), the neighborhoods match (step 918). For example, 7 neighbor matches for a 15-neighbor neighborhood can be considered a match of the neighborhoods.

If the neighbors do not match (step 902) or if the number of neighbor matches is less than the predetermined number of neighbor matches (step 906), it is determined whether the measured neighbor used in comparison is the last neighbor in the measured chunk (step 908).

-36-

If the measured neighbor is not the last measured
neighbor in the measured chunk, the next measured neighbor
is read (step 910) and the new measured neighbor is
compared with the reference neighbor already in use. If

5   the measured neighbor is the last measured neighbor in the
measured chunk, it is determined whether the reference
neighbor is the last reference neighbor in the reference
chunk (step 912). If the reference neighbor is not the
last reference neighbor in the reference chunk, the next

10   reference neighbor and the first measured neighbor are
read (step 914) and compared (step 900). If, however, the
reference neighbor is the last reference neighbor in the
reference chunk, the neighborhoods do not match (step
916).

15   FIG. 11A illustrates step 900 in FIG. 10 in detail
and shows a neighbor to neighbor comparison. The
distances d_i of an reference neighbor is compared with a
distance d_i' of a measured neighbor in step 1000. This
distance comparison is done with straight subtractions,

20   i.e. $|d_i - d_i'| = d_{idiff}$. If the distance difference d_{idiff} meets
the distance tolerance $d_{itolerance}$ (step 1002), angle ϕ_i is
compared with angle ϕ_i' (step 1004).

     Angle comparison is again done in a similar manner as
minutia angle comparison described above. Specifically,

25   angle ϕ_i matches angle ϕ_i' if $|\phi_j - \phi_j'| \leq \phi_{jtolerance}$ and $\phi_j -$
$\phi_j' \geq \phi_{jmax} - \phi_{jtolerance}$ (step 1006).

     Angles φ_i and φ_i' are then compared in step 1008.
Similarly, angle φ_i matches angle φ_i' if $|\varphi_j - \varphi_j'| \leq \varphi_{jtolerance}$
and $\varphi_j - \varphi_j' \geq \varphi_{jmax} - \varphi_{jtolerance}$ (step 1010). The neighbors match

30   (step 1012) if all parameters for a neighbor, e.g.,
distance d_i and angles ϕ_i and φ_j, match. The typical value
for $d_{itolerance}$ is 1 and the typical value for $\phi_{itolerance}$ and

$\varphi_{itolerance}$ is 6. These typical values are derived from experimentation.

If any of the parameters, i.e., distance d_i or angle ϕ_i or φ_i, do not match, the neighbors do not match (step 1014). Again, the sequence of comparison, i.e., d_i, ϕ_i and φ_i, may be modified.

FIG. 11B shows an alternative embodiment of neighbor comparison process in step 900. In this embodiment, the measured chunks are sorted in accordance with distance d_i'. Measured distances d_i' is subtracted from reference distance d_i to derive a difference d_{idiff} (step 1050). If the distance difference d_{idiff} meets a predetermined distance tolerance $d_{itolerance}$ (step 1052), angle ϕ_i is compared with angle ϕ_i' (step 1054). If angles ϕ_i and ϕ_i' match (step 1056), φ_i and φ_i' are compared in step 1058. If angles φ_i and φ_i' match (step 1060), the neighbors match (step 1062).

If measured distance d_i does not match reference distance d_i', it is determined whether the difference d_{idiff} is less than the predetermined tolerance $d_{itolerance}$ (step 1053). If the difference d_{idiff} is less than the predetermined tolerance $d_{itolerance}$, the neighbors do not match and the remaining measured neighbors are not compared. The process continues with step 912 in FIG. 10. If any of the parameters, i.e., distance d_i or angle ϕ_i or φ_i, do not match, the neighbors do not match (Step 1064).

In one embodiment, for multiple users where multiple reference fingerprints are stored in a database, if the measured fingerprint template does not match the first reference fingerprint template, the next reference fingerprint template is compared with the measured fingerprint template, as shown in FIG. 12. Multiple reference fingerprint template comparison starts with step

-38-

680.   The first reference fingerprint template and the measured fingerprint template are loaded into the RAM (one chunk at a time)(step 682).   The fingerprint templates are compared in step 684, in accordance with the template

5   comparison algorithm described above, with reference to FIGs. 8A and 8B.   Whether there is a match between the fingerprint templates is determined in step 686.   If there is a match (step 688), the process ends in step 696.   If there is no match, it is determined whether the last

10   reference fingerprint template has been used (step 690). If the last reference fingerprint template has been used, there is no match (step 692) and the process ends in step 696.   However, if the last reference fingerprint template has not been used (step 690), the next reference

15   fingerprint template is loaded to be compared with the measured fingerprint template (step 694).   In general, the number of reference templates can be stored is limited by the memory capacity.   For example, in a conventional smart card, the typical static memory available is 1 Kilobytes

20   to 16 Kilobytes.   Therefore, the number of reference templates can be stored is limited by that memory capacity.

In another embodiment for a multi-user system, the measured fingerprint template is compared with all the

25   reference templates in a similar manner as described above.   However, in this embodiment, for each template comparison, all data chunks in both the measured fingerprint template and the reference template are compared.   For example, even when a match is found for a

30   pair of fingerprint templates, e.g., more than four data chunk matches, the remaining combinations of data chunks are compared.   The number of data chunk matches for each reference fingerprint template is stored.   If the measured

-39-

fingerprint template matches more than one reference fingerprint templates, then the user producing the reference fingerprint template that gives the highest number of data chunk matches is identified as the user.

5   The algorithm in accordance with the present invention may be executed in a microprocessor having minimal capabilities, such as an 8-bit microprocessor on a conventional smart card because every parameter used in the algorithm is less than 8 bits. For example, location

10 parameters x_j and y_j are quantized to four bits each; minutia angle θ_j and angles ϕ_i and φ_i are quantized to six bits each; and distance d_i is quantized to five bits. In addition, the fingerprint template matching algorithm in accordance with the present invention does not involve

15 computational intensive processes involving for example, floating points, multiplications, and divisions. Instead, the algorithm in accordance with the present invention only uses subtractions. Hence, the algorithm can be executed by a weak microprocessor having very low speed,

20 e.g., between 1 and 10 MegaHertz, such as a microprocessor in a conventional smart card.

        Although the invention has been described with reference to particular embodiments, the description is only an example of the invention's application and should

25 not be taken as a limitation. Various other adaptations and combinations of features of the embodiments disclosed are within the scope of the invention as defined by the following claims.